Fig.1

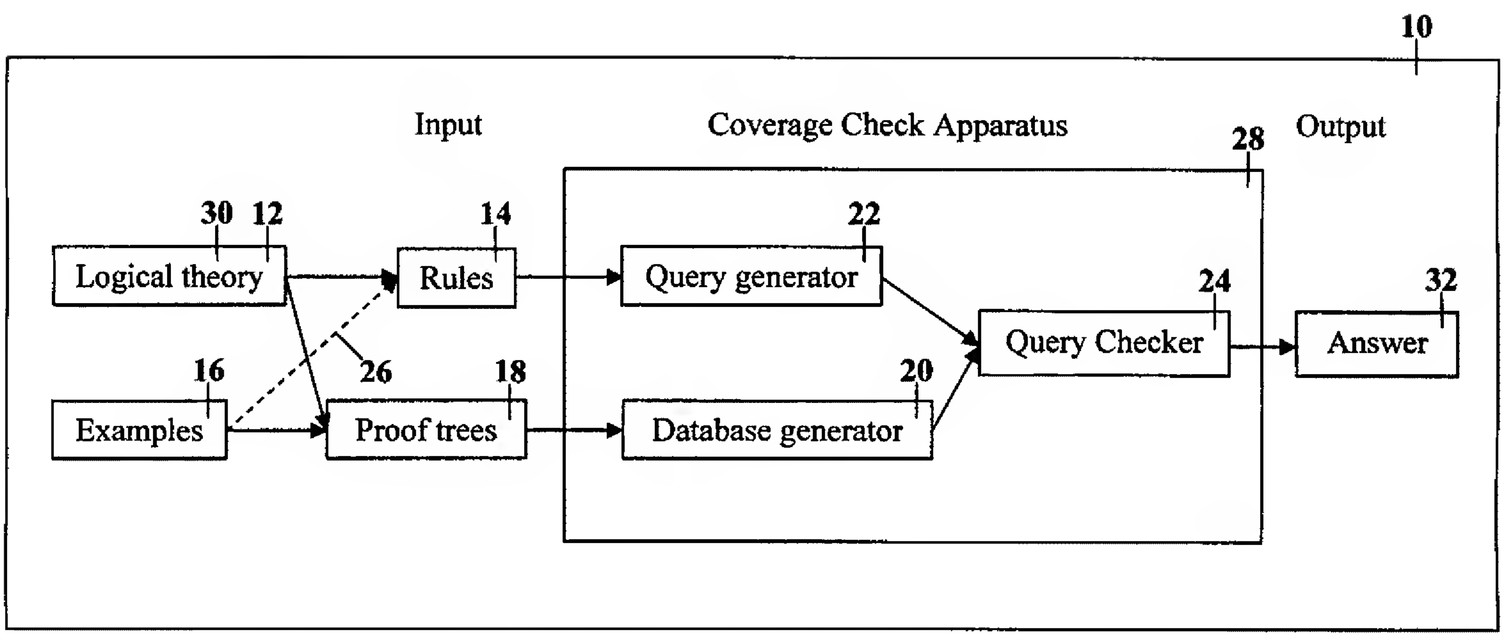
10
Input
Coverage Check Apparatus
28
Output
30
12
Logical theory
14
Rules
22
Query generator
24
Query Checker
32
Answer
16
Examples
26
18
Proof trees
20
Database generator

Fig.2

```
                                                                              30

                           33                  34                36
(c1) reward(Color,Value):- color(Color), value(Value).

                                                 29
(c2)  color(Color):- red(Color).
(c3)  color(Color):- black(Color).

(c4)  value(Value):- face(Value).
(c5)  value(Value):- numbered(Value).

(c6)  red(Color):- Color = hearts.
(c7)  red(Color):- Color = diamonds.

(c8)  black(Color):- Color = spades.
(c9)  black(Color):- Color = clubs.

(c10) face(Value):- Value = king.
(c11) face(Value):- Value = queen.
(c12) face(Value):- Value = knight.

(c13) numbered(Value):- Value = 1.
(c14) numbered(Value):- Value = 2.
(c15) numbered(Value):- Value = 3.
(c16) numbered(Value):- Value = 4.
(c17) numbered(Value):- Value = 5.
(c18) numbered(Value):- Value = 6.
(c19) numbered(Value):- Value = 7.
(c20) numbered(Value):- Value = 8.
(c21) numbered(Value):- Value = 9.
(c22) numbered(Value):- Value = 10.
```

Fig.3

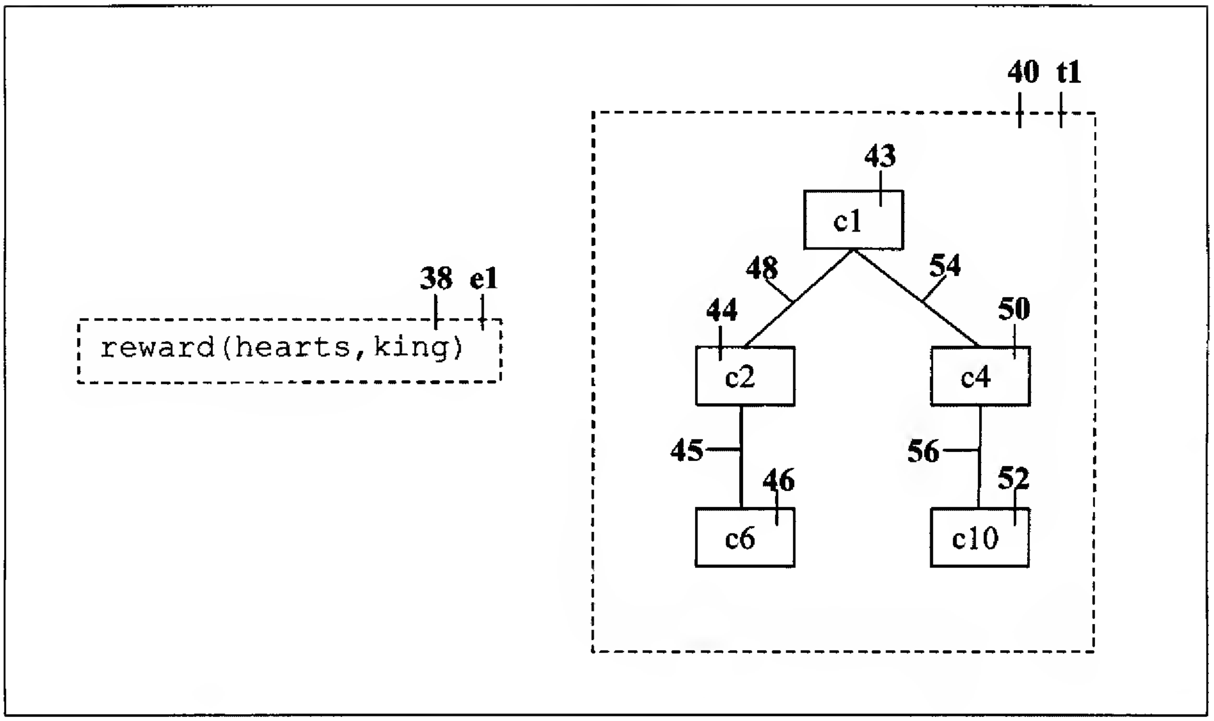

Fig.4

41

Input:
an example label e,
a proof tree T,
proof tree label t,
a set of database tables D
Output:
a set of database tables D

For each sequence $n_0, \ldots, n_k$ in the tree T, where n_0 is the root of T and n_{i+1} is a child of n_i in T, for all $0 \leq i < k$, do
Let n be a table name obtained by a function from the sequence of pairs $(c_0,1), (c_1,s_1), \ldots, (c_k, s_k)$, where c_i is the clause used in node n_i, for $0 \leq i \leq k$ and where s_i is the s_i:th child of n_{i-1}, for $0 < i \leq k$.
If there is no table named n in D, create such a table with name n and two fields, Example and Tree, and add the table to D.
Add the tuple Example = e and Tree = t to the table named n.

Fig.5

42

Table c1 42a

Example	Tree
e1	t1

Table c1_1_c2 42b

Example	Tree
e1	t1

Table c1_1_c2_1_c6 42c

Example	Tree
e1	t1

Table c1_2_c4 42d

Example	Tree
e1	t1

Table c1_2_c4_1_c10 42e

Example	Tree
e1	t1

Fig.6

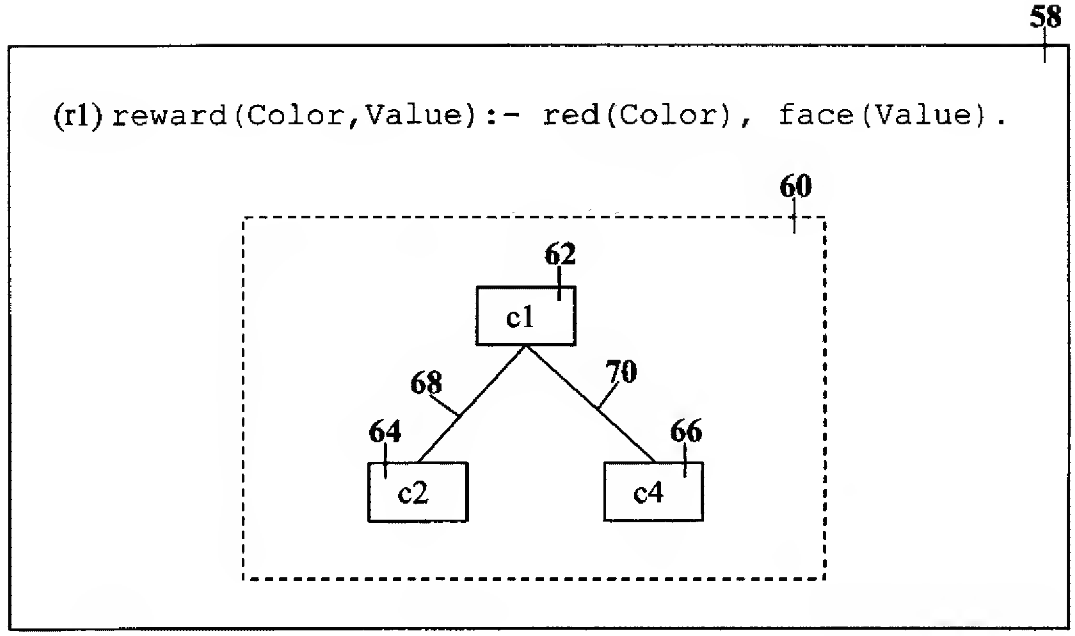

Fig. 7

22

Input:
  a partial proof tree T,
  an example label e,
Output:
  a database query Q — 72

Let D be the empty set
Let C be an empty conjunction
For each sequence $n_0, \ldots, n_k$ in the partial proof tree T, where n_0 is the root of T and n_{i+1} is a child of n_i in T, for all $0 \leq i < k$, do
  Let n be a table name obtained by a function from the sequence of pairs
  $(c_0,1), (c_1,s_1), \ldots, (c_k, s_k)$, where c_i is the clause used in node n_i, for $0 \leq i \leq k$
    and where s_i is the s_i:th child of n_{i-1}, for $0 < i \leq k$.
  Add n to D
  Add the conjunct n.Example = e to C
Let C’ = C
For each conjunct n_i.Example = e in C = (n0.Example = e) AND … AND (n_m.Example = e), where i < m, do
  Add the conjunct n_i.Tree = n_{i+1}.Tree to C’
Let Q = ‘SELECT * FROM’ + D + ‘WHERE’ + C’

Fig. 8

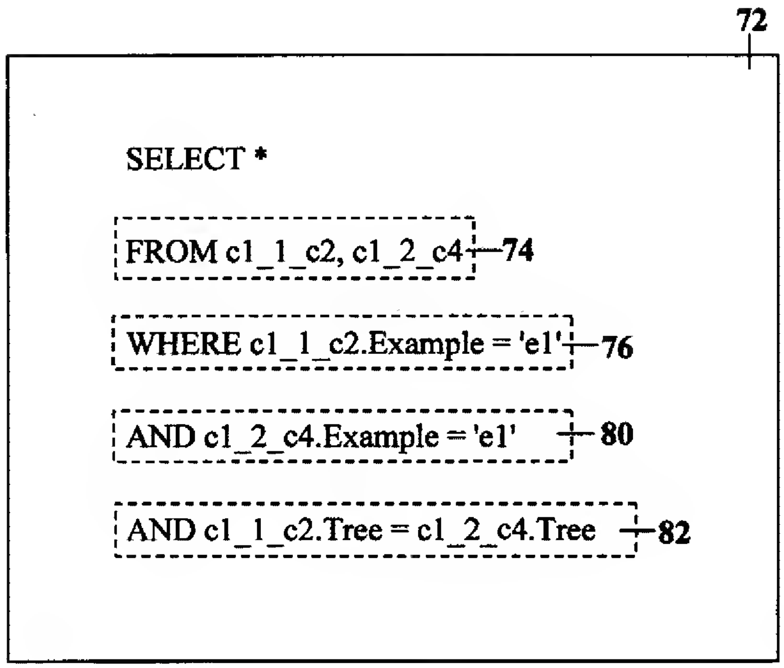
72
SELECT *
FROM c1_1_c2, c1_2_c4
74
WHERE c1_1_c2.Example = 'e1'
76
AND c1_2_c4.Example = 'e1'
80
AND c1_1_c2.Tree = c1_2_c4.Tree
82

Fig.9

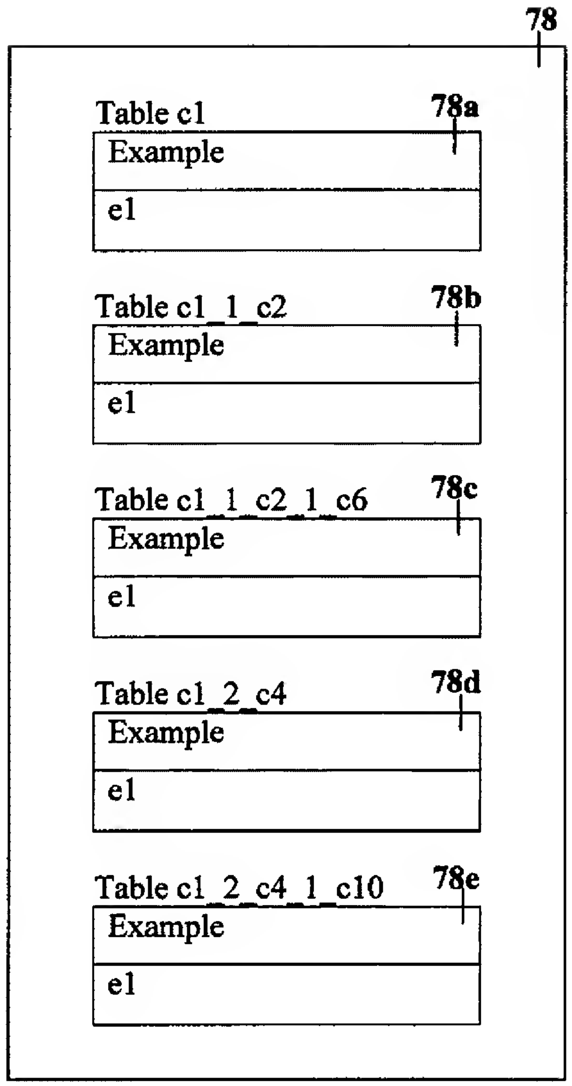
78
Table c1
78a
Example
e1
Table c1_1_c2
78b
Example
e1
Table c1_1_c2_1_c6
78c
Example
e1
Table c1_2_c4
78d
Example
e1
Table c1_2_c4_1_c10
78e
Example
e1

Fig. 10

84

```
SELECT *
FROM c1_1_c2, c1_2_c4
WHERE c1_1_c2.Example = 'e1'
AND c1_2_c4.Example = 'e1'
```

Fig. 11

86

85

87

89

```
(s1) reward(Weight,Length):-
            split_number(Weight),
            split_number(Length).
```

Fig. 12

88

```
(r2) reward(Weight,Length):-
                      Weight > 3,
                      split_number(Weight),
                      Length =< 8.2,
                      split_number(Length).
```

Fig 13.

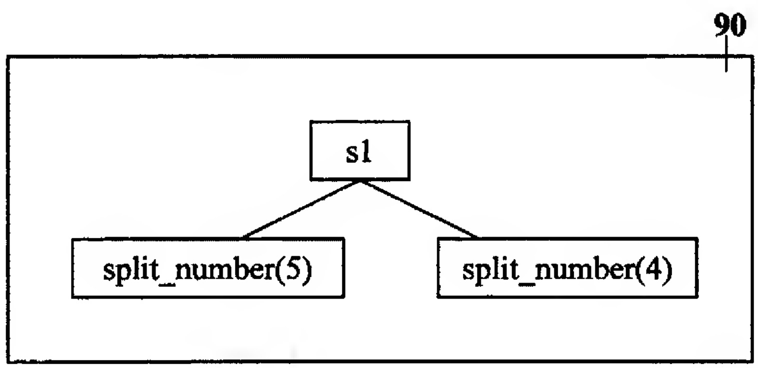

Fig. 14

92

Table s1

Example	Tree
e2	t2

Table s1_1

Example	Tree	split_number
e2	t2	5

Table s1_2

Example	Tree	split_number
e2	t2	4

Fig. 15

94

```
SELECT *
FROM s1_1, s1_2
WHERE s1_1.Example = 'e2'
AND s1_1.split_number > 3
AND s1_2.Example = 'e2'
AND s1_2.split_number <= 8,2
AND s1_1_c2.Tree = c1_2_c4.Tree
```